Discovery and Deployment of Emergent Robot Swarm Behaviors via Representation Learning and Real2Sim2Real Transfer

Connor Mattson
University of Utah
Salt Lake City, UT, USA
c.mattson@utah.edu

Varun Raveendra
University of Utah
Salt Lake City, UT, USA
varun.raveendra@utah.edu

Ricardo Vega
George Mason University
Fairfax, VA, USA
rvega7@gmu.edu

Cameron Nowzari
George Mason University
Fairfax, VA, USA
cnowzari@gmu.edu

Daniel S. Drew
University of Hawaii at Manoa
Honolulu, HI, USA
ddrew@hawaii.edu

Daniel S. Brown
University of Utah
Salt Lake City, UT, USA
daniel.s.brown@utah.edu

ABSTRACT

Given a swarm of limited-capability robots, we seek to automatically discover the set of possible emergent behaviors. Prior approaches to behavior discovery rely on human feedback or hand-crafted behavior metrics to represent and evolve behaviors and only discover behaviors in simulation, without testing or considering the deployment of these new behaviors on real robot swarms. In this work, we present Real2Sim2Real Behavior Discovery via Self-Supervised Representation Learning, which combines representation learning and novelty search to discover possible emergent behaviors automatically in simulation and enable direct controller transfer to real robots. First, we evaluate our method in simulation and show that our proposed self-supervised representation learning approach outperforms previous hand-crafted metrics by more accurately representing the space of possible emergent behaviors. Then, we address the reality gap by incorporating recent work in sim2real transfer for swarms into our lightweight simulator design, enabling direct robot deployment of all behaviors discovered in simulation on an open-source and low-cost robot platform.

KEYWORDS

Swarm Robotics; Emergent Behaviors; Representation Learning

ACM Reference Format:
Connor Mattson, Varun Raveendra, Ricardo Vega, Cameron Nowzari, Daniel S. Drew, and Daniel S. Brown. 2025. Discovery and Deployment of Emergent Robot Swarm Behaviors via Representation Learning and Real2Sim2Real Transfer. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 10 pages.

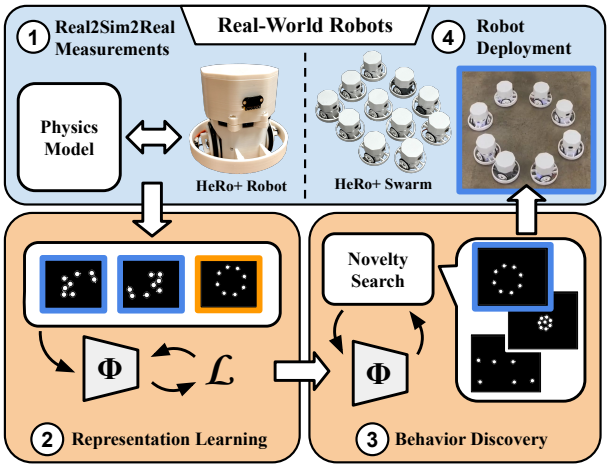


Figure 1: Real2Sim2Real Behavior Discovery via Self-Supervised Representation Learning discovers new collective behaviors for robot swarms while addressing the Sim2Real gap. **1)** Real robot measurements are implemented into a physics model in software. **2)** The model is used to generate thousands of randomly sampled behavior videos, which are used to train a representation encoder using self-supervised learning. **3)** The trained encoder is used to discover novel emergent behaviors. **4)** Interesting behaviors found in simulation can be deployed on real robots without the need for controller adjustment.

1 INTRODUCTION

Decentralized robot swarms have the potential to provide efficient, low-cost, and robust solutions to tasks such as precision agriculture [1, 7], aquatic monitoring [6, 13], search and rescue [3], construction [34], and object transportation [2, 10, 17, 45]. However, as robots become empowered with additional sensing, computation, and actuation capabilities, humans are faced with increasing cognitive complexity as they try to understand how their robots can work together to accomplish real-world goals. Researchers and practitioners currently lack the ability to fully understand and explore the design space of possible emergent swarm behaviors. In this paper, we propose and evaluate a novel pipeline for automatically discovering the set of emergent behaviors that can be achieved for swarms of robots with a set of limited capabilities. Importantly, we leverage real2sim2real techniques [30, 44] so that the behaviors we discover can be directly deployed in the real world.

Most prior research on swarm robotics focuses on optimizing swarm behaviors for specific tasks. Extensive work has been performed to optimize swarm controllers to produce target behaviors

Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

such as aggregation [4, 14–16, 18, 24, 42], circle formation [23, 41, 48], chain formation [37, 40], milling [6, 8], spatial coverage [49], self-assembly [20], segregation [35, 38] and shepherding [33], many of which have been successfully deployed on real robots. By contrast, rather than seeking a specific pre-imagined behavior, we explore the more open-ended problem of discovering the set of emergent behaviors that are possible given a particular robot swarm.

Recent work has started to address this problem by proposing approaches based on novelty search [28] for developing a taxonomy of possible emergent behaviors [9, 31, 32]. Prior work, however, is limited to simulation and does not consider the challenges involved in transferring behaviors discovered in simulation to the real world. As we show in Section 5.3, ignoring the sim2real gap results in the discovery of behaviors that only reliably work in simulation. By contrast, we seek to address the challenging sim2real gap [22, 25] in order to discover emergent swarm behaviors that are actually deployable on real robots. Furthermore, prior work on swarm behavior discovery leverages large amounts of human effort in terms of carefully hand-constructed representations of swarm behaviors. Instead, we evaluate an entirely self-supervised approach to behavior discovery, where low-dimensional representations of high-dimensional behaviors are learned without the need for human fine-tuning or physics-based hand-crafted representations.

We introduce **Real2Sim2Real Behavior Discovery via Self-Supervised Representation Learning**, a combination of sim2real transfer, behavior representation learning, and novelty search that seeks to discover the set of possible emergent swarm behaviors without the need for expensive human feedback or hand-crafted representations. As shown in Figure 1, our method starts with the real-world robots and adopts the recently proposed *Reality-to-Simulation-to-Reality for Swarms (RSRS)* process [44] to systematically approximate and implement measured robot dynamics in a simulator, and then test that behaviors produced in our adapted simulation can be accurately reproduced in the real world. Given an improved simulator, we then perform self-supervised representation learning to learn latent representations of videos of swarm behaviors. Finally, we use the learned representations to perform novelty search [28] to efficiently explore the space of emergent behaviors. By augmenting behavior discovery with the RSRS process, we enable direct deployment of all behaviors discovered in simulation to our open-source real-world HeRo+ Robots, an improved version of the educational HeRo [36] robot.

The contributions of this work can be summarized as follows: First, we propose a novel self-supervised representation learning approach for swarms based on SimCLR [11] and demonstrate that it enables quantitatively better representation learning for swarm behaviors when compared to the hand-crafted behavior representations used in prior work [9].

Second, we improve the open-source HeRo [36] robot hardware to enable accurate time-of-flight sensing, be more robust to collisions, and reduce encoder error, resulting in improved hardware for swarm robotics and a reliable way to implement inexpensive line-of-sight sensing. We make our modified robot, HeRo+, open-source for other researchers to deploy.

Third, we demonstrate the first deployment and evaluation of emergent behavior discovery for robot swarms by augmenting behavior discovery with real2sim2real calibration [44]. Inspired by prior work on computation-free swarms [17], we define a simple line-of-sight capability model and show that our method automatically discovers deployable emergent swarm behaviors for aggregation, cyclic pursuit, and dispersal. By contrast, behaviors discovered via unaligned simulations (where sensing and actuation parameters are not tuned based on real-world observations) have a much lower chance of working in the real world.

Finally, we highlight the practical considerations for deploying multi-robot systems while also paving the way for researchers to more easily discover and explore the space of emergent behaviors that are possible given a swarm of robots.

2 PROBLEM STATEMENT

Following existing nomenclature [9, 32], we define our robots as agents with a well-defined capability model, $C = \langle S, M, A \rangle$ composed of sensing (S), memory (M), and actuation (A) capabilities. In this paper, we seek to answer the following research question: **Given N robots with capabilities C, what is the complete set of emergent behaviors that can be deployed on these robots?**

We model this problem as a search for a set of emergent behaviors in a behavior space $\mathcal{B}$. The difficulty of this problem stems from the assumption that we have no direct access to this behavior space. Instead, we seek to sample and simulate swarm controllers and infer their behavioral characteristics based on the visual output of a simulator. We assume that we know the space of possible swarm controllers, $U(C)$, and the swarm's environment, $\mathcal{E}$. The controller space and environment form the input parameters for a behavior map, $\phi : U(C) \times \mathcal{E} \rightarrow \mathcal{B}$, that returns a behavior representation in the space $\mathcal{B}$. While prior work has assumed access to a known function ϕ [9], we consider the case where there is no predefined knowledge of how to represent behavior characteristics in a low-dimensional space where search can be performed.

3 METHODS

The goal of our work is to leverage machine learning to learn low-dimensional latent representations of swarm behaviors, then use that model as the basis for exploration in search of new swarm behaviors that can be deployed on real robots. Our work differs from prior work in that it learns a latent representation model in an entirely self-supervised manner and our work leverages recent work in Swarm Real2Sim2Real transfer [44], enabling direct deployment of discovered emergent behaviors to a real swarm of robots.

In the following subsections, we describe Real2Sim2Real Behavior Discovery via Self-Supervised Representation Learning which employs in-simulation representation learning (3.1), behavior exploration and discovery via novelty search (3.2), and simulator design that enables rapid and reliable real-world deployment (3.3).

3.1 Representation Learning

In order to discover new behaviors, we need a way to be able to characterize and represent different swarm behaviors. In prior work on behavior discovery, behavioral representations were explicitly hand-crafted as functions of the robots' Cartesian position and velocity [9]. However, recent advancements in representation learning enable training networks to represent high-dimensional data (images, video, etc.) as low-dimensional latent vectors that contain

encoded information about the original data. Rather than manually crafting behavioral characteristics, we study to what extent we can leverage unsupervised representation learning to create meaningful embeddings from videos of swarm behaviors.

3.1.1 Learning Paradigm. To achieve both self-supervised training and sufficient representation learning, we employ the popular Simple Framework for Contrastive Learning of Visual Representations (SimCLR) [11]. SimCLR is based on *contrastive learning*, where representations are learned by comparing and/or contrasting pairwise or triplet elements of the training data. For example, a network could learn that two elements of the data that are similar should be embedded in close proximity within the latent space (and vice versa for data that are different). In our case, SimCLR samples a swarm behavior video, $\mathbf{x}$, from our video dataset, uses two data transformations to alter the visual appearance, denoted $\widetilde{\mathbf{x}}_\mathbf{i}$ and $\widetilde{\mathbf{x}}_\mathbf{j}$, and then optimizes a network to embed $\widetilde{\mathbf{x}}_\mathbf{i}$ and $\widetilde{\mathbf{x}}_\mathbf{j}$ closer together in the latent space while considering all other elements of a batch as dissimilar from $\widetilde{\mathbf{x}}_\mathbf{i}$ and $\widetilde{\mathbf{x}}_\mathbf{j}$. This results in the following loss function for a positive pair of elements in a batch of size N,

$$\mathcal{L}_{i,j} = -\log \frac{\exp(\text{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\text{sim}(z_i, z_k)/\tau)}, \tag{1}$$

where $z_{\{i,j,k\}}$ is the latent embedding of $\widetilde{x}_{\{i,j,k\}}$, *sim* is a function that measures the similarity between two vectors (e.g., cosine similarity, L2 distance), τ is a temperature parameter, and $\mathbb{1}_{[k \neq i]}$ is a function that evaluates to 1 if and only if $k \neq i$ and evaluates to 0 otherwise. This loss objective is a cross-entropy formulation that simply seeks to maximize the likelihood that the ith and jth embeddings have the highest measure of similarity when compared to all other elements in the batch.

3.1.2 Data Augmentation. Chen et al. [11] conducted a thorough analysis of which augmentations produced the highest performing learned representation, and several other studies have thoroughly explored the benefits of pixel-based data augmentation in machine learning [5, 26, 46]. In our paper, we implement one of the highest scoring combinations of transformations studied in prior work: random crop followed by random rotation [11].

3.1.3 Encoder/Projection Architecture. Following training, we use our learned SimCLR encoder, ϕ, as a means of obtaining low-dimensional behavior representations of each swarm video. These representations are then used to determine how similar two behaviors are. We highlight the importance of correctly defined notions of similarity in the following section, where we describe how we use our encoder to search for novel emergent behaviors.

3.2 Behavior Discovery

We seek to automatically discover new behaviors, which is both a non-stationary and non-trivial objective. In particular, methods that are apt for approximating stationary functions using gradient-based approaches are unlikely to converge to solutions that truly explore the space of all possible behaviors. Instead, we follow prior work in behavior discovery [9, 31, 32] by implementing an evolutionary approach to exploration problems called *Novelty Search* [28]. Novelty search serves as a fitness mechanism that rewards representations that are different from all previously observed representations. To facilitate this, novelty search aggregates representations to a dynamic buffer, denoted B. For any newly observed representation, $b \in \mathbb{R}^d$ in d-dimensional latent space, the novelty of b is defined as

$$\text{Novelty}(b, B) = \frac{1}{k} \sum_{i=0}^{k} \text{dist}(b, B_i), \tag{2}$$

where k is the number of nearest neighbors in B to consider and B_i is the ith nearest neighbor of b in B.

In several studies, novelty search has been used to aid optimization problems by incentivizing exploration in tandem with an optimization objective [19, 27, 29]. By contrast, our sole objective is exploration and novelty, which enables us to use it as the only component of a fitness function for evolutionary search. Evolutionary search is commonly utilized in robotics literature as a means of gradient-free optimization [12]. Succinctly, evolutionary search attempts to maximize a fitness function $f(g)$ by genetically evolving populations of genomes, g, where the highest scoring genomes are more likely to mutate and survive across multiple generations. In the context of our approach, our fitness function is the novelty function (Eq. 2) and the genome is a parameterized swarm controller that can be represented as a vector $g \in U(C)$. Let the function $\mathcal{S}(g)$ denote the simulation of controller g which returns a behavior video, $\mathbf{x}$. Then, with the representation encoder, ϕ, the evolutionary optimization can be written in the form

$$\max_{g \in U(C)} \text{Novelty}(\phi(\mathcal{S}(g)), B). \tag{3}$$

As previously mentioned, the objective shown above is non-stationary in the sense that a genome will always return a higher novelty score in an earlier generation than the same genome would in a later generation. This means that solutions that have high fitness early in the search will not be as novel in subsequent generations, requiring the algorithm to test new genomes to try to diversify the behavior space. After search, the novelty buffer (B) is passed through a k-Medoids clustering algorithm and the resulting behaviors are returned to the user.

3.3 Real2Sim2Real Simulator Design (RSRS)

We augment the problem formation for behavior discovery used in prior work by explicitly targeting direct sim2real deployment for our discovered behaviors. One natural way to enable this is to simply use a high-fidelity robot simulator that can model the physics of the real world with sufficient precision. While this solution may be appropriate for methods that are directly optimizing for a specific behavior, our problem requires evaluating thousands of swarm controllers for simulations with hundreds of timesteps. We seek to instead use a simulator that is lightweight, as has been utilized in prior literature for swarm simulators that require costly search [31, 39]. Simultaneously, we do not want to neglect the dynamics of the real world, as over-simplification may produce behaviors that are infeasible for hardware deployment. Therefore, we require a strategic simulator design approach that enables both lightweight evaluation and closes the reality gap.

Reality-to-Simulation-to-Reality for Swarms (RSRS) [44] is a new simulator design paradigm that leads to more feasible and reliable real-world swarm deployments. The main idea behind RSRS is that the robots in simulation are *less* capable than they actually are

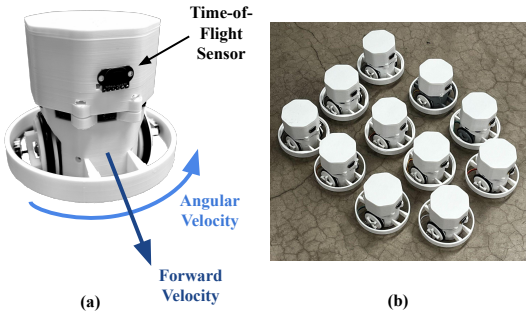


Figure 2: HeRo+ Robots: We deploy newly discovered behaviors on a fleet of HeRo+ robots. **(a)** A single HeRo+ robot uses unicycle commands to locomote and time-of-flight sensing to detect other robots. Our open-source robot design is mostly 3D-printed and costs approximately $80-USD, making it an extremely low-cost option for swarms research. **(b)** Our swarm of 11 HeRo+ Robots, 8 of which are used in this study.

in the real world. At first, this may appear as a weakness, as our problem is defined with respect to the capabilities of real world robots. Although the dynamics of the real world are difficult to efficiently represent in simulation, we can approximate these real-world uncertainties by exaggerating their impact on our robots. For example, the collision dynamics between two robots cannot be perfectly modeled in a lightweight simulator, but real robot collisions can guide our approach to simulator design through informative observations. For example, for the observation that "these robots cannot reliably slide past each other if they collide head-on," this perhaps indicates that the friction coefficient should be adjusted to ensure that the simulator does not attempt to find behaviors that exploit collisions. We follow the four steps described in the RSRS process [44]: **1)** Measure the capabilities and dynamics of real-world robots, **2)** Implement the measurement data into the robot simulator, **3)** Run experiments in simulation, modify robots and simulator as needed, **4)** Perform experiments on real robots, modify robots and simulator as needed.

In particular, it should be noted that steps 3 and 4 involve iterative refinement to the simulator and robots in order to reproduce behaviors in the real world. RSRS lays out a simple *if-else* approach for modifying the simulator and robots. If it is less expensive to upgrade the robots to improve reliability than to modify the simulator, upgrade the robots. Otherwise, make the simulator more realistic and take more measurements on real robots. RSRS has been previously shown to be effective when optimizing sim2real transfer for a specific desired behavior [44]. By contrast, our approach expands the use of this design paradigm to broadly enable Real2Sim2Real transfer for open-ended behavior discovery. We discuss the details of these steps and discuss the modifications we made to our robots and simulator in Sections 4.1 and 4.2.

4 EXPERIMENTS

We demonstrate the efficacy of our methods by deploying new behaviors, discovered in simulation, on real swarm robots. Following our 4-stage RSRS process, we first model the interactions between our low-cost HeRo+ robots (Figure 2) in simulation using measurements obtained from the real world (4.1-4.2). Second, we generate a video dataset that is used to train a SimCLR encoder using self-supervised learning (4.3), and employ evolutionary behavior discovery to evolve and discover unique behaviors that diversify the set of embedded features in the encoder (4.4). Finally, the behaviors discovered in simulation are deployed on our robots in the real world (4.5). Videos, code, and open-source hardware designs are available on our project webpage [1].

4.1 Robot Hardware

4.1.1 Kinematics and Controllers. Our experiments consider a homogeneous swarm of 8 robots modeled in 2D with unicycle kinematics, where the ith robot is controlled at time t with a forward velocity, $v_{i,t}$, and angular velocity, $\omega_{i,t}$. Our robots only receive a binary observation $h_{i,t} \in \{0, 1\}$ from a line-of-sight sensor and use this signal as the condition for an *if-else* style controller of the form $[u_{v,0}, u_{\omega,0}, u_{v,1}, u_{\omega,1}] \in U(C)$. Though the values of this controller are shared by all the agents in the swarm, the velocity of different agents may vary based on the agent's individual observation state,

$$(v_{i,t}, \omega_{i,t}) = \begin{cases} (u_{v,0}, u_{\omega,0}) & \text{if } h_{i,t} \text{ is } 0, \\ (u_{v,1}, u_{\omega,1}) & \text{otherwise.} \end{cases} \tag{4}$$

Because the 4-tuple controller representation is time-invariant, it allows for control of the behavior of the entire swarm, for an arbitrary simulation horizon, with just four scalar values. These four values, constrained by the robots' practical velocity limits, form the space of possible controllers, $U(C)$, where we search for behaviors as described in Section 3.2.

4.1.2 Improvements to Robot Hardware. The HeRo Robot [36] is an open-source, low-cost, 3D-printed robot with two-wheeled differential drive actuation. These robots act using only local observations and are effectively decentralized. However, for ease of deployment and control, the robots are connected to a centralized ROS server, allowing for full swarm emergency stop, synchronized start, and wireless controller updates.

Our first emergent behavior test on these robots was to attempt to get them to perform a "Cyclic Pursuit" behavior (all robots form a circle and rotate about the center) using already discovered unicycle controllers from prior work [31, 44]. Recall from Section 3.3 that iteration for RSRS can take two forms: Hardware Upgrades and Software Upgrades. Using our observations from the real world, we close the reality gap from a hardware perspective by improving the HeRo hardware with the following features:

Bump Shield: Based on our observations, the most significant hindrance to emergent behavior was that collisions between robots often resulted in actuation difficulties, especially when the contact was chassis-to-wheel (causing a direct force on the servo motors and wheels). For behaviors like cyclic pursuit, the robots may bump into each other during formation, which would effectively halt some robots in collision, resulting in a pile-up. Rather than attempting

[1] https://sites.google.com/view/swarmdiscovery-with-rsrs/home

to carefully model the difference between head-on and chassis-to-wheel collisions, we found that augmenting robots with a 3D-printed bump shield was an inexpensive way to allow the robots to collide with each other without actuation faults.

Time-of-Flight Sensing: By default, the HeRo robot uses 8 IR sensors, evenly spaced around the robot's circumference, to sense its surroundings. We tested our binary controller with just the forward-facing IR sensor and found that the sensor could not detect robots at a distance greater than 25 cm and reported false negatives 50% of the time at the 25 cm range, which inhibited the ability of the robots to correctly sense each other and form together into a behavior. While these errors can be measured and modeled in simulation, we found that sensing reliability was critical to behavior formation, necessitating a hardware upgrade. We implement an inexpensive upgrade by adding a single laser-ranging Time-of-Flight sensor (VL53L1X, see Figure 2a) that can detect other robots up to 2 m away and is significantly more reliable, with almost no false negatives when the robots are driving at low speeds.

Encoder Feedback: Lastly, we found that the original position of the gear-driven encoders on the HeRo robot resulted in frequent reading errors that affected the reliability of low-level PID control. While it would be difficult to efficiently model this type of uncertainty in simulation, moving the encoder outside the wheels of the robot and connecting it with a directly-driven shaft significantly reduced the error frequency.

We found that these hardware improvements greatly increase the reliability of swarm controllers. We call this RSRS-improved HeRo robot, the **HeRo+ robot**. All CAD models and a bill of materials have been made available in our supplemental materials.

4.1.3 Environment: The robots are placed in a 170x142 cm arena. The four walls of the arena are each 5 cm tall and were deliberately designed so as to be shorter than the TOF sensor on the HeRo+ robots, preventing them from detecting the walls so that anything in line-of-sight can be assumed to be another agent. The environment also has 3 x 4 = 12 grid initialization points that are roughly centered in the arena. For simulated controllers, the 8 robots are randomly assigned to one of the 12 starting locations and randomly oriented in the range $[0, 2\pi]$ (following similar instantiation as other robot swarm studies [18]). When running tests on the real-robots, we replicate as closely as possible the same starting positions and orientations used in simulation.

4.2 Robot Simulator

Our HeRo+ robots have a maximum linear velocity of 20 cm/s and a maximum angular velocity of 3 rad/s. However, we noticed that at high-speeds, the robots did not have sufficient time to sense other robots; follow-up tests indicated that the robots had near-perfect sensing at max speeds of 9 cm/s and 1.6 rad/s, which is what we chose to implement in simulation so that our robots could switch velocity commands with sufficient reaction time. In the context of RSRS design, this was a much cheaper way to bridge the reality gap than upgrading the robots with increased sensing frequency or attempting to model the sensing reliability as a function of velocity.

Based on the interactions between robots and the environment we also observed that friction between the robots and the wall was a major aspect that impacted the robot's behavior. Notably, in the default simulator, the collisions are frictionless, which allow robots to slide along walls and other robots with ease. In the real world, however, even though our bump shields allowed for safe collisions, those collisions still involve friction. Therefore, we ran intentional collision tests on our robots and manually approximated the friction coefficient of robot-to-robot and robot-to-wall collisions until our simulator visually matched our observations in the real world. This approach successfully prevents the robots from relying on frictionless interactions to form emergent behaviors.

4.3 Representation Learning

4.3.1 Training Data. Given our RSRS simulator, we randomly sample unicycle controllers from the constrained space of control velocities measured on the real world robots to create 6000 training videos. To reduce the size of the training data, we render all simulation in greyscale and we also resize the original simulation resolution from 513x426 to 64x64. Each simulation runs for a fixed duration of 600 timesteps (dt=0.1). To improve processing and training speed, we sub-sample each video to form an input of size (3, 64, 64), where the channel dimension represents 3 greyscale images evenly spaced over the last 300 timesteps of simulation, which we qualitatively found to capture the final converged emergent behavior.

For the random input transformations, we apply a random crop that scales the image in the range $[0.6, 1.0]$ with a 1:1 aspect ratio, a horizontal flip is then applied with probability p=0.5. For random rotation, we select a random rotation angle $\theta \in \{0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}\}$ and rotate the axes of the video around the image center.

4.3.2 Deep Learning. We instantiate our representation model, ϕ, as a pretrained ResNet18 [21] model with a modified final output of size 128. All but the last layer of the ResNet represent the encoder, and a final 2-layer MLP is used to project the embedding into a space where the loss is applied. The latent embedding is a vector of size 512, which reflects the ResNet's default layer size. As recommended in prior work [11], we only use the encoder part of the network for downstream evaluation. We train for 100 epochs with a mini-batch size of 1000 videos, following the large-batch recommendations of SimCLR [11]. Each video is passed through the random crop and random rotation transformations to produce a total of 1000 video pairs. For training, we use the NX-Ent loss [11] from Equation 1 paired with the LARS optimizer [47] with a learning rate of $1.17 = (0.3 \times \text{BatchSize}/256)$ and weight decay of 1.5×10^{-7}. All other hyperparameters follow the original SimCLR implementation [11].

4.3.3 Baseline: Hand-Crafted Metrics. We compare our representation learning to the set of hand-crafted behavior features used by prior work on robots of this same capability model (i.e., differential drive with line-of-sight sensing) [9]. Each metric captures a scalar-valued characteristic of the collective motion of the agents including average speed, angular momentum, radial variance, scatter, and group rotation. When concatenated, the five metrics form a behavior representation $b \in \mathbb{R}^5$.

4.4 Behavior Discovery

Inspired by prior work [31], we use a tournament-style genetic algorithm to evolve our controllers under the objective function in Equation 3. We start with an initial population of 50 controllers

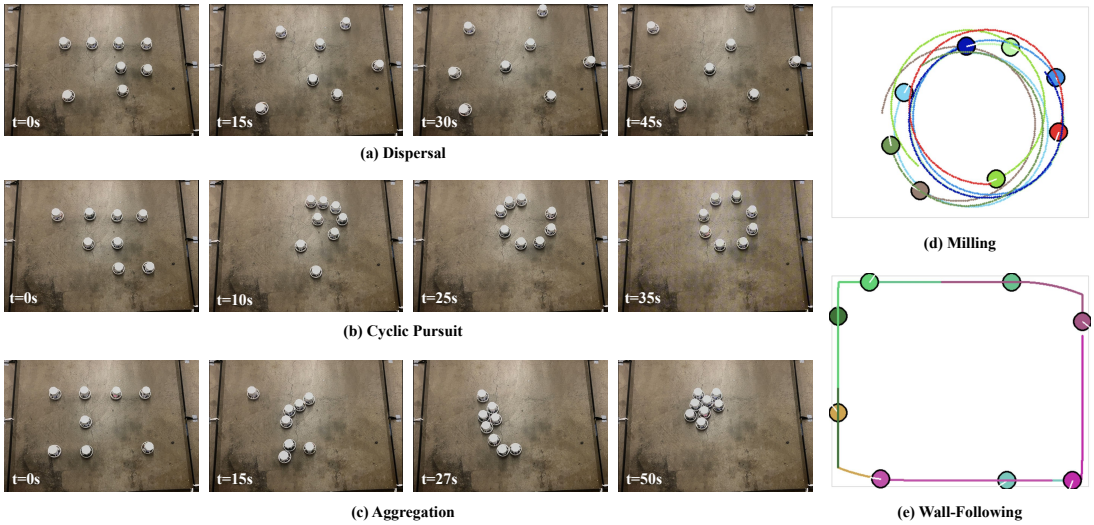


Figure 3: Discovered Behaviors. Behaviors **a-c** were automatically discovered and deployed on HeRo+ Robots using Real2Sim2Real (RSRS) Representation Learning for Behavior Discovery. Behaviors **d-e** were discovered in simulation for a simulator without RSRS improvements, but could not be deployed to the real world and were not discovered when RSRS was added. **(a) Dispersal**: Agents cover the environment by moving away from other robots. **(b) Cyclic Pursuit**: Agents form a circular chain and revolve around the center. **(c) Aggregation**: Agents cluster together in the middle of the environment. **(d) Milling**: All agents revolve around the centroid of the swarm, but they do not form a perfect circle like cyclic pursuit. **(e) Wall-Following**: Agents slide along the walls and trace the outer edge of the environment.

randomly sampled from the controller space and run the evolutionary search for 100 generations, each with a population of 50 genomes. At the end of every generation, the resulting behavior videos from each controller are passed through the encoder (or evaluated with the baseline metrics) and saved to the buffer for use in novelty search. Following prior work [31], we compute novelty (Equation 2) with respect to the 15 nearest-neighbors in the buffer and use a same crossover rate of 0.7 and a mutation rate of 0.15. Novelty search results in a buffer of 5000 controllers. The representations of these behaviors are then clustered using k-Medoids with k=10, resulting in 10 behaviors (medoids) from the search that are selected for evaluation in the real world.

4.5 Real-World Deployment

As we did not explicitly filter our behavior space before clustering, it is likely that some behaviors found in behavior discovery will show agents crashing into walls, not moving, or not producing a collective behavior; we refer to these behaviors as *Random* behaviors. We note that after behavior discovery, it is up to the discretion of the human to determine which behaviors they want to try to reproduce on the robots in the real world. For a fair evaluation, we reproduce all non-random behaviors in the real world to assess the success of our RSRS simulator design.

For each experiment, we record whether or not the behavior was successfully reproduced in the real world and any adjustments to initial conditions or controllers that were required to produce the behavior. It should be noted that our assessment of behavior reproduction is only considered with respect to the swarm's high-level behavior. As there will always be uncertainties in the real world that cannot be modeled in simulation, we do not expect perfect agent-level sim2real alignment and we do not measure how accurately each robot follows its individual simulated trajectory. Rather, our goal is to bring our simulator close enough to reality that swarm-level behavior can be reliably reproduced on real robots.

5 RESULTS

Our results average 3 runs of behavior discovery for both the baseline hand-crafted metrics and our self-supervised learned representation. Across both methods, k-Medoids returns 30 non-random controllers for deployment and evaluation in the real world. We show that our automated discovery can detect emergent behaviors of Cyclic Pursuit (Cyc.), Aggregation (Agg.) and Dispersal (Disp.) that can be deployed directly into the real world (Figure 3 a-c). We first analyze the performance of our representation learning, then discuss the behaviors that were discovered and successfully deployed onto real-world robots. We also highlight results from an additional study which supports the inclusion of RSRS design in our approach, where we show two other behaviors, Milling and Wall-Following (see Figure 3 d-e), that are also discovered by a naive behavior search approach that does not use RSRS. While interesting, these emergent behaviors cannot be deployed into the real world, whereas all the emergent behaviors discovered via our approach are directly deployable on our real robots.

5.1 Representation Alignment

Although we did not require any labeled data for training, we evaluate how well our encoder performs using a set of 500 labeled testing videos. We first compute a set of representation-specific confusion matrices (Figure 4), which indicate how often

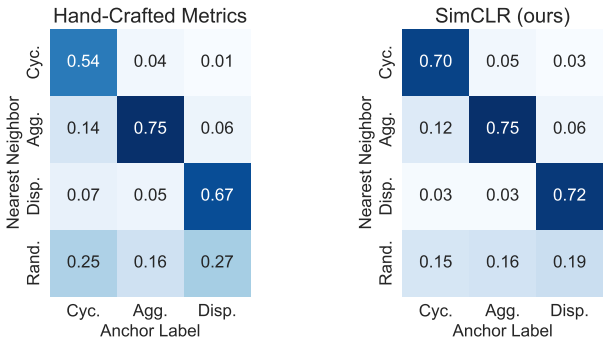


Figure 4: Representation Confusion Matrices for **(left)** the baseline hand-crafted representations and **(right)** the self-supervised representations on a held-out set of 500 labeled test videos. Classes along the horizontal axis indicate the labeled class of an anchor behavior and the values along the vertical axis display the frequency with which a behavior from the anchor's class (on-diagonal) was embedded closer to the anchor when compared with a behavior from one of the other classes (off-diagonal). Larger diagonal values indicate stronger within-class correlation in the embedded representation space.

different examples of the same emergent behavior were embedded closer together than a different emergent behavior. To evaluate the quality of our learned latent representation, we utilize a triplet, $\langle a, p, n \rangle$, consisting of an anchor (a), positive (p), and negative (n) example, where a and p have the same label and n is a different label than both a and p. Under a correct learned representation, we would expect the embedded representations a and p to be more similar in the embedding space than the embedding of n when compared to a or p. We adapt the triplet distance formation to create a representation confusion matrix, where we evaluate all valid triplets in the labeled testing data defined by $\{\langle a, p, n \rangle \subset X_{\text{test}} \mid a_{\text{label}} = p_{\text{label}}, a_{\text{label}} \neq n_{\text{label}}\}$, and then test to see if $dist(a, p) < dist(a, n)$. We conduct this test for both the baseline and self-supervised representation methods. We evaluate the static hand-crafted metrics with a single evaluation and the learned representation as the averages of 3 SimCLR training runs.

Figure 4 shows our findings for emergent behavior representation alignment, where we see that both methods display a clear correlation within-behavior as shown by the high values on the diagonal. We find that, although the baseline metrics were specifically crafted to reflect swarm behavior characteristics, the self-supervised capabilities of SimCLR are able to perform with slightly better or equal accuracy than the baseline for all non-random behaviors (+16% cyclic pursuit, +0% aggregation, +5% dispersal), indicating that *the learned model can sufficiently capture behavioral semantics using only self-supervised representation learning*, and that it outperforms the hand-crafted approach.

We also evaluate the representation of the labeled data from a qualitative perspective by visualizing the embeddings in 2D space using t-SNE dimensionality reduction [43]. The hand-crafted metrics (Figure 5a) show a clear ability to distinguish random behaviors from cyclic pursuit and aggregation. However, the hand-crafted

Table 1: Discovered Behavior Frequency for novelty search with k=10 behaviors returned in each trial. Results are averaged across 3 runs and displayed alongside the standard error.

	Behavior Representation		
Discovered Behaviors	Hand Crafted [9]	Triplet Learning [31]	Self-Supervised (ours)
Aggregation	4.33 ± 0.66	3.66 ± 0.66	2.0 ± 0.0
Cyclic Pursuit	0.66 ± 0.33	2.0 ± 0.0	1.0 ± 0.57
Dispersal	0.0 ± 0.0	0.0 ± 0.0	2.0 ± 0.57
Random	5.0 ± 1.0	4.33 ± 0.66	5.0 ± 1.0
Total Unique (excl. Random)	1.6 ± 0.33	2.0 ± 0.0	**2.6 ± 0.33**

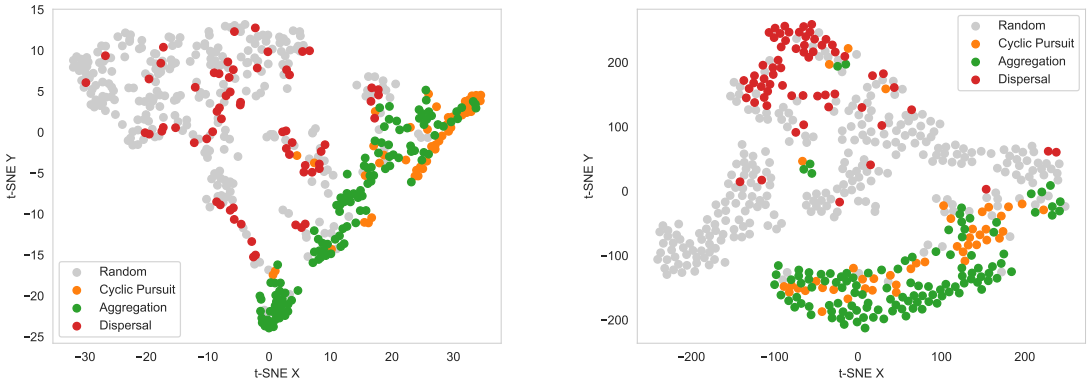
Table 2: Real2Sim2Real (RSRS) Experiments where our self-supervised method is compared with and without the RSRS [44] simulator improvements in 3 trials of behavior discovery. Although the default simulator discovered additional behaviors, they could not be produced in the real world. Legend: (✓) Behavior discovered and one-shot deployed on real robots, (✓) behavior discovered and deployed with 2-3 attempts, (✗) behavior discovered, but not successfully deployed, (–) behavior not discovered. Multiple entries per cell indicate that the behavior was returned from behavior discovery more than once.

	No RSRS (Sim. Default)			RSRS		
Behavior	Trial 1	Trial 2	Trial 3	Trial 1	Trial 2	Trial 3
Aggregation	✓✗	–	✗✗✗	✓✓	✓✓	✓✓
Cyclic Pursuit	✗	✓	✓✗	✓	✓✓	–
Dispersal	–	✓✓	–	✓	✓✓✓	✓✓
Milling	–	✗	✗✗	–	–	–
Wall Following	✗	✗✗	–	–	–	–

representation poorly differentiates between dispersal and random controllers, which supports the findings of our quantitative data (Figure 4a) that show that hand-crafted dispersal representations are confused for random representations 27% of the time. Notably, in Figure 5b, the learned embedding does still correctly differentiate aggregation and cyclic-pursuit from random, but performs with 8% better accuracy when separating dispersal from random. Importantly, in both representation confusion and t-SNE evaluations, our self-supervised model is being evaluated on data that was not seen during training, indicating a strong ability to generalize.

5.2 Discovery and Deployment

We run behavior discovery 3 times for both the baseline metrics and our learned representation. Each time, a set of 10 controllers is output and categorized by behavior. The frequency of returned behaviors is shown in Table 1. We find on average that the number of behaviors extracted from each method appears to correlate very closely with how the behaviors were distributed in the t-SNE visualization, with dispersal never being discovered in the baseline method, as one might hypothesize based on Figure 5a. Though the other behaviors vary consistently, it is also worth noting that both methods return the same number of random behaviors on average.



(a) Hand-Crafted Behavior Metrics [9] **(b) Self-Supervised Representation Learning (ours)**

Figure 5: t-SNE Embeddings for the **(a)** baseline hand-crafted representations and the **(b)** self-supervised representations on a held-out set of 500 labeled test videos. Qualitatively, the hand-crafted baseline is able to densely associate cyclic pursuit behaviors, but fails to differentiate dispersal from random behaviors. In our approach, behavior features have more variance, but are more closely associated with behaviors from the same class, notably dispersal, which is embedded more distinctly from random behaviors than in the baseline case.

From the 60 controllers examined in simulation, exactly 30 (15 from each method) were identified as non-random emergent behaviors. We directly deployed the same controllers found in simulation on our HeRo+ robots in the real world. From the 30 non-random controllers discovered in the RSRS simulator, we were able to one-shot reproduce **70% (20)** of them in the real world and **90% (27)** were successfully reproduced within 3 attempts, without any controller or initialization adjustment. Our methods show the potential for sim2real transfer in tandem with behavior discovery using the RSRS method. We validate this further in a thorough ablation study where we examine the importance of each improvement to our simulator by deploying behaviors discovered under simulators with incomplete RSRS measurements.

5.3 Importance of RSRS in Swarm Deployment

To demonstrate the importance of RSRS in our approach, we ran 3 additional trials of our self-supervised method on the original simulator, with almost no RSRS improvements. To ensure a fair comparison that has the potential to produce deployable behaviors, we implement the physical dimensions of the original HeRo robots and upper-bound the controller with the robot's maximum forward and angular velocities, but we do not include any of our measurements for friction, reasonable speeds for sensing, or the augmented geometry from the bump shield. Our experiments (Table 2) show that a total of 18 non-random controllers were discovered, including two behaviors that were not discovered in our RSRS behavior discovery: wall following and milling. Of the 18 controllers returned from this ablation experiment, only 22% (4) of the behaviors could be one-shot reproduced on the robots and 27% (5) were successfully reproduced within 3 attempts. Compared to the default simulator, *the inclusion of RSRS improves the one-shot success rate by 48% and the three-shot success rate by 63%.*

Notably, neither wall following nor milling were reproduced successfully, indicating that the RSRS behavior discovery did not erroneously miss these behaviors—rather, these behaviors are artifacts of an imperfect simulator and are not achievable with the real capabilities of our HeRo+ robots. For wall following, real-world friction prevents agents from sliding along the walls of the environment. Milling could not be achieved in the real world because it leads to many head-on collisions between robots, which cannot easily slip past one another as they can in the unrefined simulator.

6 CONCLUSION AND FUTURE WORK

We present Real2Sim2Real Behavior Discovery via Self-Supervised Representation Learning and show the successful real-world deployment of emergent behaviors discovered in simulation. We also demonstrate that purely self-supervised learned behavioral representations can be used in place of burdensome hand-crafted metrics and outperform the hand-crafted metrics in terms of emergent behavior representation and discovery. In the future, we are excited to apply our approach to novel swarm capability models, including limited communication capabilities. We are also excited to study the effect of the environment ($\mathcal{E}$) on the possible emergent swarm behaviors. Finally, we believe that behavior discovery has the potential to enable swarm roboticists to discover novel ways of using robots to accomplish interesting and useful tasks, enabling the confident deployment of automatically discovered behaviors into real-world scenarios. Future work should study specific tasks where novelty and discovery can improve how we solve problems with scalable swarms of low-cost, limited-capability robots.

ACKNOWLEDGMENTS

This work was conducted in the Aligned, Robust, and Interactive Autonomy (ARIA) Lab at the University of Utah. ARIA Lab research is supported in part by the NSF (IIS-2310759, IIS2416761), the NIH (R21EB035378), ARPA-H, and the ARL STRONG program.

REFERENCES

[1] Dario Albani, Tiziano Manoni, Arikhan Arik, Daniele Nardi, and Vito Trianni. 2019. Field Coverage for Weed Mapping: Toward Experiments with a UAV Swarm. In *Bio-inspired Information and Communication Technologies*, Adriana Compagnoni, William Casey, Yang Cai, and Bud Mishra (Eds.). Vol. 289. Springer International Publishing, Cham, 132–146. https://doi.org/10.1007/978-3-030-24202-2_10

[2] Muhanad H. Mohammed Alkilabi, Aparajit Narayan, and Elio Tuci. 2017. Cooperative object transport with a swarm of e-puck robots: robustness and scalability of evolved collective strategies. *Swarm Intelligence* 11, 3 (Dec. 2017), 185–209. https://doi.org/10.1007/s11721-017-0135-8

[3] Ross D. Arnold, Hiroyuki Yamaguchi, and Toshiyuki Tanaka. 2018. Search and rescue with autonomous flying robots through behavior-based cooperative intelligence. *Journal of International Humanitarian Action* 3, 1 (Dec. 2018), 18. https://doi.org/10.1186/s41018-018-0045-4

[4] Farshad Arvin, Khairulmizam Samsudin, Abdul Rahman Ramli, and Masoud Bekravi. 2011. Imitation of Honeybee Aggregation with Collective Behavior of Swarm Robots. *International Journal of Computational Intelligence Systems* 4, 4 (June 2011), 739–748. https://doi.org/10.1080/18756891.2011.9727825

[5] Philip Bachman, R Devon Hjelm, and William Buchwalter. 2019. Learning representations by maximizing mutual information across views. *Advances in neural information processing systems* 32 (2019).

[6] Florian Berlinger, Melvin Gauci, and Radhika Nagpal. 2021. Implicit coordination for 3D underwater collective behaviors in a fish-inspired robot swarm. *Science Robotics* 6, 50 (Jan. 2021), eabd8668. https://doi.org/10.1126/scirobotics.abd8668

[7] Timo Blender, Thiemo Buchner, Benjamin Fernandez, Benno Pichlmaier, and Christian Schlegel. 2016. Managing a Mobile Agricultural Robot Swarm for a seeding task. In *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*. 6879–6886. https://doi.org/10.1109/IECON.2016.7793638

[8] Daniel S Brown, Sean C Kerman, and Michael A Goodrich. 2014. Human-swarm interactions based on managing attractors. In *Proceedings of the 2014 ACM/IEEE international conference on Human-robot interaction*. 90–97.

[9] Daniel S. Brown, Ryan Turner, Oliver Hennigh, and Steven Loscalzo. 2018. Discovery and Exploration of Novel Swarm Behaviors Given Limited Robot Capabilities. In *Distributed Autonomous Robotic Systems*, Roderich Groß, Andreas Kolling, Spring Berman, Emilio Frazzoli, Alcherio Martinoli, Fumitoshi Matsuno, and Melvin Gauci (Eds.). Vol. 6. Springer International Publishing, Cham, 447–460. https://doi.org/10.1007/978-3-319-73008-0_31

[10] Jianing Chen, Melvin Gauci, and Roderich Groß. 2013. A strategy for transporting tall objects with a swarm of miniature mobile robots. In *2013 IEEE International Conference on Robotics and Automation*. 863–869. https://doi.org/10.1109/ICRA.2013.6630674 ISSN: 1050-4729.

[11] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*. PMLR, 1597–1607.

[12] Stephane Doncieux, Nicolas Bredeche, Jean-Baptiste Mouret, and Agoston E Eiben. 2015. Evolutionary robotics: what, why, and where to. *Frontiers in Robotics and AI* 2 (2015), 4.

[13] Miguel Duarte, Vasco Costa, Jorge Gomes, Tiago Rodrigues, Fernando Silva, Sancho Moura Oliveira, and Anders Lyhne Christensen. 2016. Evolution of Collective Behaviors for a Real Swarm of Aquatic Surface Robots. *PLOS ONE* 11, 3 (March 2016), e0151834. https://doi.org/10.1371/journal.pone.0151834

[14] Luigi Feola, Antoine Sion, Vito Trianni, Andreagiovanni Reina, and Elio Tuci. 2023. Aggregation Through Adaptive Random Walks in a Minimalist Robot Swarm. In *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, Lisbon Portugal, 21–29. https://doi.org/10.1145/3583131.3590485

[15] Simon Garnier, Christian Jost, Jacques Gautrais, Masoud Asadpour, Gilles Caprari, Raphaël Jeanson, Anne Grimal, and Guy Theraulaz. 2008. The Embodiment of Cockroach Aggregation Behavior in a Group of Micro-robots. *Artificial Life* 14, 4 (Oct. 2008), 387–408. https://doi.org/10.1162/artl.2008.14.4.14400

[16] Melvin Gauci, Jianing Chen, Tony J Dodd, and Roderich Groß. 2014. Evolving aggregation behaviors in multi-robot systems with binary sensors. In *Distributed Autonomous Robotic Systems: The 11th International Symposium*. Springer, 355–367.

[17] Melvin Gauci, Jianing Chen, Wei Li, Tony J Dodd, and Roderich Groß. 2014. Clustering objects with robots that do not compute. In *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. 421–428.

[18] Melvin Gauci, Jianing Chen, Wei Li, Tony J. Dodd, and Roderich Groß. 2014. Self-organized aggregation without computation. *The International Journal of Robotics Research* 33, 8 (July 2014), 1145–1161. https://doi.org/10.1177/0278364914525244

[19] Jorge Gomes, Paulo Urbano, and Anders Lyhne Christensen. 2013. Evolution of swarm robotics systems with novelty search. *Swarm Intelligence* 7 (2013), 115–144.

[20] Roderich Gross, Michael Bonani, Francesco Mondada, and Marco Dorigo. 2006. Autonomous Self-Assembly in Swarm-Bots. *IEEE Transactions on Robotics* 22, 6 (Dec. 2006), 1115–1130. https://doi.org/10.1109/TRO.2006.882919

[21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[22] Nick Jakobi, Phil Husbands, and Inman Harvey. 1995. Noise and the reality gap: The use of simulation in evolutionary robotics. In *Advances in Artificial Life: Third European Conference on Artificial Life Granada, Spain, June 4–6, 1995 Proceedings 3*. Springer, 704–720.

[23] Shin-Young Jung, Daniel S Brown, and Michael A Goodrich. 2013. Shaping couzin-like torus swarms through coordinated mediation. In *2013 IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, 1834–1839.

[24] Serge Kernbach, Dagmar Häbe, Olga Kernbach, Ronald Thenius, Gerald Radspieler, Toshifumi Kimura, and Thomas Schmickl. 2013. Adaptive collective decision-making in limited robot swarms without communication. *The International Journal of Robotics Research* 32, 1 (Jan. 2013), 35–55. https://doi.org/10.1177/0278364912468636

[25] Sylvain Koos, Jean-Baptiste Mouret, and Stéphane Doncieux. 2012. The transferability approach: Crossing the reality gap in evolutionary robotics. *IEEE Transactions on Evolutionary Computation* 17, 1 (2012), 122–145.

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems* 25 (2012).

[27] Joel Lehman and Kenneth O Stanley. 2010. Efficiently evolving programs through the search for novelty. In *Proceedings of the 12th annual conference on Genetic and evolutionary computation*. 837–844.

[28] Joel Lehman and Kenneth O Stanley. 2011. Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary computation* 19, 2 (2011), 189–223.

[29] Joel Lehman and Kenneth O Stanley. 2011. Evolving a diversity of virtual creatures through novelty search and local competition. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*. 211–218.

[30] Vincent Lim, Huang Huang, Lawrence Yunliang Chen, Jonathan Wang, Jeffrey Ichnowski, Daniel Seita, Michael Laskey, and Ken Goldberg. 2021. Planar robot casting with real2sim2real self-supervised learning. *arXiv preprint arXiv:2111.04814* (2021).

[31] Connor Mattson and Daniel S. Brown. 2023. Leveraging Human Feedback to Evolve and Discover Novel Emergent Behaviors in Robot Swarms. In *Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, Lisbon Portugal, 56–64. https://doi.org/10.1145/3583131.3590443

[32] Connor Mattson, Jeremy C. Clark, and Daniel S. Brown. 2023. Exploring Behavior Discovery Methods for Heterogeneous Swarms of Limited-Capability Robots. In *2023 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*. 163–169. https://doi.org/10.1109/MRS60187.2023.10416780

[33] Anil Özdemir, Melvin Gauci, and Roderich Groß. 2017. Shepherding with robots that do not compute. In *Artificial Life Conference Proceedings*. MIT Press One Rogers Street, Cambridge, MA 02142-1209, USA journals-info …, 332–339.

[34] Kirstin Petersen, Radhika Nagpal, Justin Werfel, et al. [n.d.]. TERMES: An Autonomous Robotic System for Three-Dimensional Collective Construction.

[35] Paulo Rezeck, Renato M. Assunção, and Luiz Chaimowicz. 2021. Flocking-Segregative Swarming Behaviors using Gibbs Random Fields. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. 8757–8763. https://doi.org/10.1109/ICRA48506.2021.9561412 ISSN: 2577-087X.

[36] Paulo Rezeck, Héctor Azpúrua, Maurício F. S. Corrêa, and Luiz Chaimowicz. 2023. HeRo 2.0: a low-cost robot for swarm robotics research. *Autonomous Robots* 47, 7 (Oct. 2023), 879–903. https://doi.org/10.1007/s10514-023-10100-0

[37] Paulo Rezeck and Luiz Chaimowicz. 2022. Chemistry-Inspired Pattern Formation With Robotic Swarms. *IEEE Robotics and Automation Letters* 7, 4 (Oct. 2022), 9137–9144. https://doi.org/10.1109/LRA.2022.3190638

[38] Vinicius G. Santos, Anderson G. Pires, Reza J. Alitappeh, A. F. Rezeck Paulo, C. A. Pimenta Luciano, Douglas G. Macharet, and Chaimowicz Luiz. 2020. Spatial segregative behaviors in robotic swarms using differential potentials. *Swarm Intelligence* 14, 4 (12 2020), 259–284.

[39] Shay Snyder, Kevin Zhu, Ricardo Vega, Cameron Nowzari, and Maryam Parsa. 2023. Zespol: A Lightweight Environment for Training Swarming Agents. In *Proceedings of the 2023 International Conference on Neuromorphic Systems*. 1–5.

[40] Valerio Sperati, Vito Trianni, and Stefano Nolfi. 2011. Self-organised path formation in a swarm of robots. *Swarm Intelligence* 5 (2011), 97–119.

[41] David St-Onge, Carlo Pinciroli, and Giovanni Beltrame. 2018. Circle formation with computation-free robots shows emergent behavioural structure. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 5344–5349.

[42] J. Timmis, A. R. Ismail, J. D. Bjerknes, and A. F. T. Winfield. 2016. An immune-inspired swarm aggregation algorithm for self-healing swarm robotic systems. *Biosystems* 146 (Aug. 2016), 60–76. https://doi.org/10.1016/j.biosystems.2016.04.001

[43] Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, 11 (2008).

[44] Ricardo Vega, Kevin Zhu, Sean Luke, Maryam Parsa, and Cameron Nowzari. 2023. Simulate Less, Expect More: Bringing Robot Swarms to Life via Low-Fidelity

Simulations. http://arxiv.org/abs/2301.09018 arXiv:2301.09018 [cs, eess].

[45] Sean Wilson, Aurélie Buffin, Stephen C. Pratt, and Spring Berman. 2018. Multi-robot replication of ant collective towing behaviours. *Royal Society Open Science* 5, 10 (Oct. 2018), 180409. https://doi.org/10.1098/rsos.180409

[46] Qizhe Xie, Zihang Dai, Eduard Hovy, Thang Luong, and Quoc Le. 2020. Unsupervised data augmentation for consistency training. *Advances in neural information processing systems* 33 (2020), 6256–6268.

[47] Yang You, Igor Gitman, and Boris Ginsburg. 2017. Large batch training of convolutional networks. *arXiv preprint arXiv:1708.03888* (2017).

[48] Kevin Zhu, Connor Mattson, Shay Snyder, Ricardo Vega, Daniel S Brown, Maryam Parsa, and Cameron Nowzari. 2024. Spiking Neural Networks as a Controller for Emergent Swarm Agents. In *2024 International Conference on Neuromorphic Systems (ICONS)*. IEEE, 319–326.

[49] Anıl Özdemir, Melvin Gauci, Andreas Kolling, Matthew D. Hall, and Roderich Groß. 2019. Spatial Coverage Without Computation. In *2019 International Conference on Robotics and Automation (ICRA)*. 9674–9680. https://doi.org/10.1109/ICRA.2019.8793731 ISSN: 2577-087X.